Current Research Topic In Software Engineering

A PROJECT REPORT

Submitted by

MD. Mithun Ahamed

Id: 13-96937-2

Under the guidance of

DR. Dip Nandi

in partial fulfillment for the award of the degre of

Master of Science

in

Computer Science

of

School of Computer Science

American International University of Bangladesh

Dhaka, Bangladesh

July 2013

TABLE OF CONTENTS PAGES

ABSTRACT.. 2

KEYWORDS.. 3

CHAPTER NO 1. Introduction.. 4

CHAPTER NO 2. Aspects of Software Engineering.. 5

2.1 Current Practices.. 5

CHAPTER NO 3. Artificial Intelligence and Software Engineering...................... 6

3.1 Intersections between AI and SE... 6

3.2 Knowledge-Based Software Engineering............................... 6

CHAPTER NO 4. Natural Language Processing.. 7

4.1 Efficiency... 7

4.2 Multimodality.. 7

CHAPTER NO 5. Applications of Data mining in software engineering............... 8

5.1 Mining software engineering data... 8

5.2 Clustering & Text mining... 8

CONCLUSION.. 9

ACKNOWLEDGEMENT.. 10

REFERENCES... 11

ABSTRACT

Software engineering processes are complex, and the related activities often produce a large number and variety of artifacts. This paper describes some factors that Reseaech in Software engineering firm or industry.There are much more topics and fields in software engineering senario. Research in SE was concerned with supporting human beings to develop better software faster. Today several research directions of both disciplines come closer together and are beginning to build new research areas. but this paper i will short overview some important issues which are handeling very hardly and consciously.

KEYWORDS

Data mining, software engineering, applications, Software assurance, Natural Language Processing , Artificial Intelligence, Clustering , Text mining, Multimodality, Knowledge-Based Software Engineering.

CHAPTER NO 1

Introduction:

Software systems are inherently complex and difficult to conceptualize. This complexity compounded by intricate dependencies and disparate programming paradigms, slows development and maintenance activities, leads to faults and defects and ultimately increases the cost of software. Most software development organizations develop some sort of processes to manage software development activities.

However, as in most other areas of business, software processes are often based only on hunches or anecdotal experience, rather than on empirical data. It is notoriously difficult to construct conventional software systems systematically and timely (Sommerville, 2001), with up to 20% of industrial development projects failing.

Consequently, many organizations are 'flying blind' without fully understanding the impact of their process on the quality of the software that they produce. This is generally not due to apathy about quality, but rather to the difficulty inherent in discovery and measurement. Every firm or industry have research and development sector to improve software engineering important fields.

CHAPTER NO 2

Aspects of Software Engineering:

The discipline of SE was born 1968 at the NATO conference in Garmisch-Partenkirchen, Germany [52, 71] where the term "SE crisis" was coined. Its main concern is the efficient and effective development of high-qualitative and mostly very large software systems. The goal is to support software engineers and managers in order to develop better software faster with (intelligent) tools and methods. Since its beginning, several research directions developed and matured in this broad field.
Software Engineering is concerned with the acquisition, definition, management, monitoring, and controlling of software development projects as well as the management of risks emerging during project execution. The research for Software Design & Architecture advances techniques for the development, management, and analysis of (formal) descriptions of abstract representations of the software system as well as required tools and notations.

2.1 Current Practices:

Software engineering practices many,I mention this paper some essential fields from them. Artificial Intelligence and Software Engineering, Natural Language Processing, Applications of Data mining in software engineering, Software Assurance, Software design, Software development, Software quality.

CHAPTER NO 3

Artificial Intelligence and Software Engineering:

Why is AI interesting for researchers from SE? It can provide the initial technology and first (successful) applications as well as a testing environment for ideas. The inclusion of research supports the enabling of human-enacted processes and increases user acceptance. AI Technology can help to base the overall SE method on a concrete technology, providing sufficient detail for the initial method description, and through the available reference technology clarifying the semantics of the respective method.

3.1 Intersections between AI and SE:

While the intersections between AI and SE are currently rare, they are multiplying and growing. First points of contact emerged from the application of techniques from one discipline to the other Systematic software development (including Requirements Engineering (RE), Engineering of Designs (DE), or source code (CE) or project management (PM) methods help to build intelligent systems while using advanced data analysis techniques. Knowledge Acquisition (KA) techniques help to build EF and intelligent ambient systems like Domain Modeling (DM) techniques support the construction of requirements for software systems and product lines.

3.2 Knowledge-Based Software Engineering:

SE is a highly dynamic field in terms of research and knowledge, and it depends heavily upon the experience of experts for the development and advancement of its methods, tools, and techniques. For example, the tendency to define and describe "best practices" or "lessons learned" is quite distinctive in the literature. As a consequence, it was the SE field where an organization, the EF, was introduced that was explicitly responsible to systematically deal with experience.

CHAPTER NO 4

Natural Language Processing:

A fundamental difference between NLP systems and conventional software is the incompleteness property, since current language processing techniques can never guarantee to provide all and only the correct results, the whole system design is affected by having to take this into account and providing appropriate fallbacks.

4.1 Efficiency:

Human users are very demanding (Shneiderman, 1997) reports that system response times
4*s* can render a system unacceptable. It is also debated resources, lack of trust in non-in-house components, or the inability to install or integrate existing software. Price or licensing concerns also play a role. It is argued here that software engineering techniques can improve overall productivity of researchers after some little initial investment.

4.2 Multimodality:

A language engineer applying the same parser to investigate the discourse structure of 18th
century novels does not encounter the same challenges as her colleague trying to apply it to speech dialogs. Different modalities have their own idiosyncrasies, and it is difficult to factor out all of them, but this is necessary because there is a trend toward multi-modal systems, and intra-system reuse requires a high degree of adaptability.

CHAPTER NO 5

Applications of Data mining in software engineering:

Software engineering activities generate a vast amount of data that, if harnessed properly through data mining techniques, can help provide insight into many parts of software development processes. Although many processes are domain and organization – specific, there are many common tasks which can benefit from such insight, and many common types of data which can be mined.

5.1 Mining software engineering data:

Data mining techniques have been applied in the context of software engineering,
Association rules and frequent patterns-- Zimmermann et al. (2005) have developed the Reengineering of Software Evolution (ROSE) tool to help guide programmers in performing maintenance tasks. The goals of ROSE are to:

1 suggest and predict likely changes.
2 prevent errors due to incomplete changes.
3 detect coupling undetectable by program analysis.

5.2 Clustering & Text mining:

Dickinson et al (2001) examine data obtained from random execution sampling of
Instrumented code and focus on comparing procedures for filtering and selecting data, each of which involves a choice of a sampling strategy and a clustering metric. They find that identifying failures in groups of execution traces, clustering procedures are more effective than simple random sampling adaptive sampling from clusters was found to be the most effective sampling strategy.
Text mining is an area of data mining with extremely broad applicability. Rather than requiring data in a very specific format (e.g., numerical data, database entries, etc.), text mining seeks to discover previously unknown information from textual data.

CONCLUSION

We have identified some importance why software engineering is a good fit for data mining, Artificial Intelligence, Knowledge-Based Software Engineering, Agent-Oriented Software Engineering, Natural Language Processing, Multimodality, Applications of Data mining in software engineering, Clustering & Text mining, Targeting software Tasks, Software Assurance Framework for Software, Software Security.

I have also discussed the importance about software engineering Research, And motivation that future research in this domain is likely to, focus on increased automation and greater simplicity.

ACKNOWLEDGEMENT:

I would like to thank Dr.Dip Nandi for reviewing an earlier version of this script and suggesting valuable improvements to my approach. I have received invaluable support from several individuals and groups for accessing and interpreting the real life data that was used to validate my research. I appreciate his interest in my dissertation. I thank him for all the time and attention. His assistance was vital in disseminating results, as well as engaging in collaborations and travel related to my research. He gave me independence in pursuing my research topic, and guidance in addressing the expectations of the program.
This research was partly supported by the Dr.Dip Nandi, Head of the Master of Sciense in Computer Science Department, American International University of Bangladesh. And thanks to all of my friends and class mates who's are helped me to improve my papers.

REFERENCES:

1.Basili V. R. Caldiera G., and Rombach H. D. "Experience Factory," in *Encyclopedia of Software Engineering*, vol. 1, J. J. Marciniak, Ed. New York: John Wiley & Sons, 1994.

2. Birk A., Surmann D., and Althoff K. D., "Applications of knowledge acquisition in experimental software engineering," presented at 11th European Knowledge Acquisition Workshop (EKAW'): Knowledge Acquisition,Modeling, and Management, Berlin, Germany, 1999.

3. Michail A. "Data mining library reuse patterns using generalized association rules," presented at Proceedings of the 2000 International Conference on Software Engineering (ICSE 2000), New York, 2000.

4. Partridge D., *Artificial intelligence and software engineering : understanding the promise of the future*. Chicago: Glenlake Pub. Co. Fitzroy Dearborn Publishers, ISBN 1-57958-062-9, 2000.

5. Aurum A., *Managing software engineering knowledge*. Berlin: Springer, ISBN: 3540003703, 2003.

6. Christodorescu, M. Jha, S. and Kruegel, C. (2007) 'Mining specifications of malicious behavior', in *Proceedings of the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*.

7. Cubrani_c, D. and Murphy, G.C. (2004) 'Automatic bug triage using text classification', in *Proceedings of the 16th International Conference on Software Engineering & Knowledge Engineering*.

8. Dickinson, W., Leon, D. and Podgurski, A. (2001) 'Finding failures by cluster analysis of execution profiles', in *Proceedings of the 23rd International Conference on Software Engineering*.

9. Following IEEE format.

10. Castro J., Kolp M., and Mylopoulos J., "Towards requirements- driven information systems engineering: the Tropos project," *Information Systems*, vol. 27, 2002.

11. Rombach H. D. and Ulery B. T., "Establishing a measurement based maintenance improvement program: lessons learned in the SEL," University of Maryland, College Park, Md. 1989.

12 Ruhe G. and Bomarius F., "Proceedings of Learning software organizations (LSO): methodology and applications," presented at 11th International Conference on Software Engineering and Knowledge Engineering, SEKE'99, Kaiserslautern, Germany, 1999.

13. Basili V. R., *Quantitative evaluation of software methodology*. College Park, Md.: University of Maryland, 1985.

14. Althoff K.-D., *Evaluating Case-Based Reasoning Systems: The Inreca Case Study*. Postdoctoral Thesis (Habilitationsschrift). Dept. of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany, 1997.

15. Nick M. M., *Building and Running Long-Lived Experience- Based Information Systems*. PhD Thesis. Dept. of Computer Science, University of Kaiserslautern, Kaiserslautern, to be submitted in 2004.

16. Henninger S. "Developing domain knowledge through the reuse of project experiences" *SIGSOFT Software Engineering Notes*, vol. Aug. 1995.